CLAIMS

1. N+1 P-P (parallel program, hereinafter "P-P") module based on a single machine environment, which comprises:

N+1 P-P branch programs module (N≧1), which is run by operating the N+1 P-P branch programs which have an object code independent structures, in the way of time division (without the need of time division operation under the structure of parallel computers), for making a transmission and consistency of the P-P data under the supports of three classes of sequence-net instructions (or subroutines) for reading P-P data, writing P-P data, and making P-P data consistency in said P-P branch programs; and

a managing program module, for supporting the suspension status, ready status, and running status of the P-P branch programs in response to information from the P-P branch programs.

2. The N+1 P-P module of claim 1, further comprising a suspension-processing module, for processing the suspension status, ready status and running status of the current P-P branch programs.

3. The N+1 P-P module of claim 1, further comprising a P-P entering & exit management module, for initializing the P-P for each application and processing the exit of the P-P in response to information from a time-division- managing module.

4. The N+1 P-P module of claim 1, wherein the N+1th P-P branch program of P-P module executes a P-P data sequence, which is represented by a data consistency operation.

5. The N+1 P-P module of claim 4, wherein a token consistency operation of N+1th branch program corresponds to the P-P data, which is made consistency one by one directly by the control of the N+1th program, wherein if the P-P data has been written by other P-P branch program, then the consistency of the P-P data is executed, otherwise the N+1th branch program is in suspension status, after the execution of the P-P data's consistency, the P-P data read by other P-P branch program is valid.

6. The N+1 P-P module of claim 5, wherein, when P-Ps are embed and child P-Ps are called, the call-permission of the child P-P and the P-P data sequence are sequenced in the N+1th branch program.

7. The N+1 P-P module of claim 5, wherein, the N+1 programs include three kinds of P-P instructions (or subroutines) for reading the data, writing the data, and making the data in consistency respectively, and the P-P instructions (or subroutines) have the capability for setting, detecting, and processing the tokens, and also have the capability for supporting the transmissions of P-P data and the synchronization of P-P branch programs.

8. An operational method of N+1 P-P branch programs ($N \geqq 1$) based on single machine environment, which comprises the following procedures:

initializing the N+1 P-P branch programs,

determining whether the P-P is terminated, and checking the ON/OFF switch flag of each P-P branch program, wherein:

if a switch flag of all P-P branch programs is OFF, which indicates the P-P is terminated, the record for showing the termination of the P-P and the manner for connecting with external programs are processed,

once the switch flag of one P-P branch program is ON, the suspension status for branch programs is checked, wherein,

if the check result is that all P-P branch programs are in suspension status, which indicates that though the execution of some P-P branch programs is not finished, the P-P branch programs can not be executed,

finding out the reasons that the P-P branch programs can not be executed,

processing the exiting of the P-P branch programs based on the reasons,

if the check result is that more than one P-P branch program is in ready status, enter the P-P branch programs queuing modules,

selecting a P-P branch program which is in ready status,

selecting a P-P branch program, loading its parameters, and

running the P-P branch program which is selected.

9. The N+1 P-P operation method of Claim 8, wherein, the initializing procedure comprises the loading of parameters of P-P branch programs and the resetting of the flag zone the P-P.

10. The N+1 P-P operation method of Claim 9, wherein, the loading of the parameters of the P-P branch program comprises:

loading the flag ON for the N+1 branch programs of the P-Ps,

setting an entry address for each P-P branch programs,

setting a data initial values for respective registers, and

resetting a P-P flag zone and a data flag of the P-P branch program.

11. The N+1 P-P operation method of Claim 8, wherein the P-P branch program comprises a subroutine for writing data, which comprises the following procedures:

performing the writing operation for a P-P data,

checking the flag "had been invalid for consistency" of the P-P data,

if the flag is valid (indicating that the P-P data had been made the data consistency, as the consistency flag is invalid, the consistency operation had not been operated actually to make the consistency of the P-P branch program entering into the suspension status),

the status bit of N+1th P-P branch program of the ready/suspension flag of P-P is changed from the suspension status to the ready status,

if the flag is invalid, no specific operation is executed,

establishing the flag "consistency valid" and allowing the consistency P-P branch program to perform the consistency operation to the data, and

exiting write data subroutine.

12. The P-P operation method of Claim 8, wherein the P-P branch program comprises a consistency data subroutine, which comprises the following procedures:

checking the "consistency valid" flag of this data, wherein,

if the "consistency valid" flag of this data is valid, the consistency operations of data and token are executed,

checking whether the N bits "had been read invalid" flag of this data is valid,

if the "had been read invalid" flag is valid, then the relevant P-P branch programs are changed to the ready status from the suspension status based on the content of the flag "had been read invalid",

if the "had been read invalid" flag of this data is invalid, no specific operation is proceeded,

the data consistency subroutine is terminated and returned,

if the "consistency valid" flag of this data is invalid, the P-P branch program is suspended,

setting the "had been invalid in consistency" flag,

proceeding the suspension process of the current N+1th branch program, saving the current running situation of the N+1th P-P branch program to be used when returning the P-P branch program, and

exiting the P-P branch program, and keeping the P-P branch program in suspension status and quitting from the returning port, in order to re-select a new P-P branch program.

13. The P-P operation method of Claim 8, wherein the P-P branch program comprises a read data subroutine, which comprises the following procedures:

checking the "read valid" flag of the data,

if the "read valid" flag of the data is valid,

then reading the data,

returning the read data subroutine,

if the "read valid" flag of the data is invalid,

in accordance with the branch program of the read data subroutine, the N bits "had been read invalid" flags of are set as "had been read invalid" respectively,

keeping the current P-P branch program in suspension status, saving the current running situation of the P-P branch program so that the current P-P branch program may be continued from the suspension point, and

the P-P branch program exits and turns to branch suspension & exit module in order to re-select a new P-P branch program.

14. A P-P call-instruction including the entry addresses of N+1 P-P branch programs, which comprises:

a call-instruction, for calling a P-P entering & exit management module

which is used to process and acquire the entry addresses of the parallel call of the branch programs,

a time-division-managing module for managing the operations of multiple branch programs, wherein the time-division-managing module selects and runs a first branch program from the multiple branch programs, manages the exiting and returning of the first branch program, and selects and runs a second branch program from the multiple branch programs; the time-division-managing module returns to the P-P entering & exit management module until all the multiple branch programs is run.

15. The time-division-managing module of Claim 14, wherein the time-division managing module manages the "suspension status", "ready status", and "running status" of the multiple branch programs in response to return information from the N+1 P-P branch programs in suspension status.

16. A combination structure of P-P call-instructions and call-permission-instructions, which comprises:

a precondition for executing the P-P call-instructions is that the call-permission-instructions are executed first,

the call-instructions and the call-permission-instructions are located in different branch programs respectively, and

the call-permission-instructions and the P-P data sequence (in the N+1th branch program) are sequenced simultaneously.